Identifying Patterns in Computational Thinking Problem Solving in Early Primary Education

Ivica BOTIČKI[a], Petar KOVAČEVIĆ[a], Danica PIVALICA[a] & Peter SEOW[b]
[a]*Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia*
[b]*National Institute of Education, Nanyang Technological University, Singapore*
*ivica.boticki@fer.hr

Abstract: This paper presents a study on computational thinking problem solving where first grade primary students engage in five groups of computational thinking tasks. The tasks are related to curriculum topics of language, science and mathematics and are implemented in form of a web application. Throughout the five tasks groups, students complete tasks covering the computation concepts of sequence, algorithms, recognition and removal of unnecessary steps, object properties and problem tasks. The focus of the data analysis presented in this paper is on identifying the computational thinking tasks across all five task groups where students were least successful and identifying patterns of task completion done by the students. For these least successful tasks, the correct and the incorrect completion patterns were examined. The results indicate that CT tool scaffolds serve as a mechanism through which students explore problems via trial and error and come to their own creative solutions through problem exploration.

Keywords: computational thinking, primary education, elementary education, problem solving, usage patterns

1. Introduction

The definition of computational thinking is broad since the problems to be solved using CT are not necessarily mathematically well-defined and completely analyzable problems. One definition assumes that computational thinking tackles the problems that can be solved by designing a system around it (Wing, 2011), while the other state that anything that can be solved using the abstraction process, which is done by deciding on which details to highlight and which to ignore, can be considered a form of computational thinking (Wing, 2008). Brennan and Resnick propose a framework in which they define computational thinking via the three dimensions: *computational concepts*, *computational practices* and *computational perspectives* (Brennan & Resnick, 2012). Computational concepts are the fundamental concepts of many programing languages and computing in general, such as sequences, loops, operators, data etc.

Kazimoglu et al. argue that computational thinking can be characterized by its core five skills: *problem solving*, *building algorithms*, *debugging*, *simulation* and *socializing*. They elaborate on the definition by building a custom game framework with problem tasks for practicing the core skills. However, it is to be noted that they developed and tested their framework on first year university students and relied on student feedback as the primary source of data (Kazimoglu, Kiernan, Bacon, & MacKinnon, 2012). In their research of computational thinking assessment, Roman-Gonzalez et al. developed a CT test based on computational concepts and, to a lesser extent, computational practices (Roman-Gonzalez, Perez-Gonzalez, & Jimenez-Fernandez, 2017). The test was conducted on a sample of 1,251 Spanish students from 24 schools, from fifth to tenth grade. They compared the results to other standardized psychological tests such as the Primary Mental Abilities (PMA) battery and the RP30 problem-solving test, and found high correlation between the CT test and the standardized problem-solving test and a small or moderate correlation between CT and other student abilities.

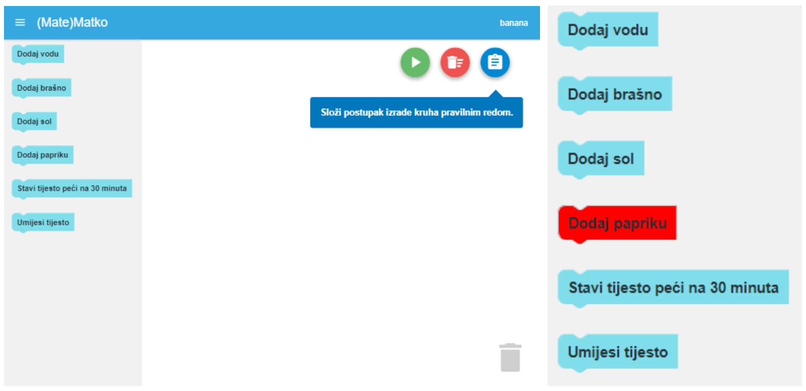
By further exploring the notion of computational thinking (CT) concepts and processes students engage in when completing CT tasks, this paper adds on the study on the use of computational thinking in early primary school in Croatia by Boticki, Pivalica & Seow (2018) in which a custom computational thinking tool Matko was used to measure the performance of 23 first grade primary school students in Croatia 7 to 8 years old completing computational thinking tasks covering the curricular topics of mathematics, language and science. The results of the study by Boticki, Pivalica & Seow (2018) present data on how well the students solved CT tasks according to the type of task they engaged in and correlations of these results with the students' prior skills and knowledge. The analysis found a strong relationship between students' mathematics skills and his or her success rate on all CT tasks, even the ones that had no connection with the Mathematics subject, and that were focused on Science or Croatian language. Language skills proved to be limiting for some students across almost all CT tasks, with the Science subject task where the goal was to define objects' properties by describing an animal, being the most affected in the analysis.

This paper aims to further the analysis presented in the previous authors' study by identifying patterns in the most difficult computational thinking tasks completed by the students.

2. Matko - A Computational Thinking System for Early Primary School Learners

2.1 System Overview

Matko is a CT tool built as part of a computational thinking project in Croatia in form of a block-based visual environment in which students drag-and-drop blocks into a scripting pane to build a solution (Figure 1). Such an environment is inspired by similar tools that were used in primary school settings (Wilson & Moffat, 2010). It approaches learning programming and the underlying computational concepts such as sequence or objects in an interactive way suitable for children (Figure 1).



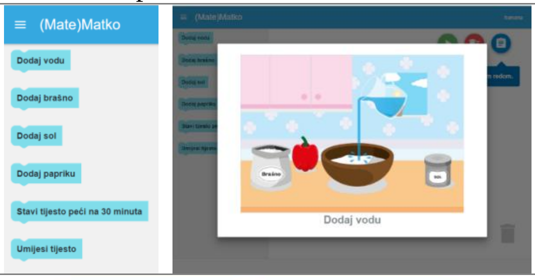
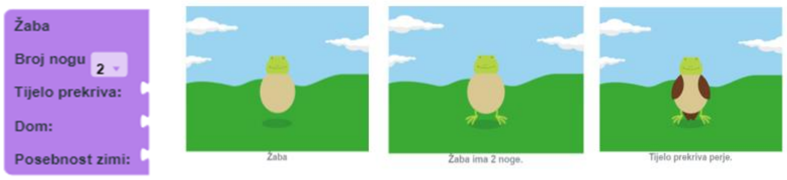
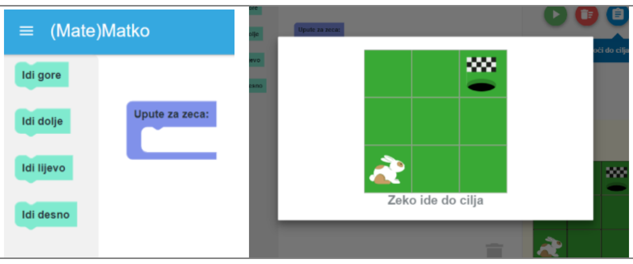
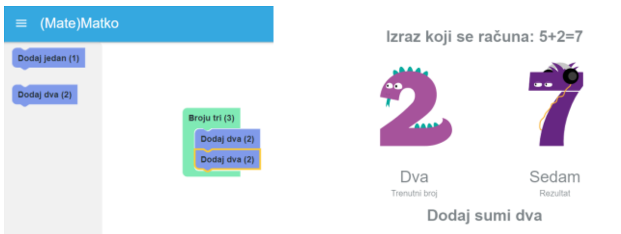
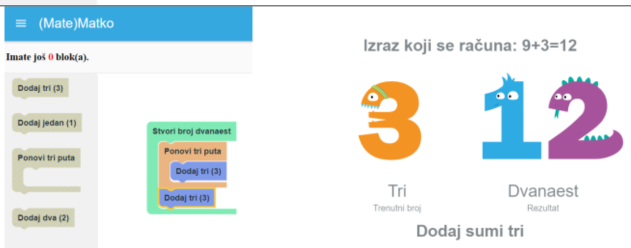
Složi postupak izrade kruha pravilnim redom.

Figure 1. Left - user interface of the CT tool presented in this study; right: enlarged toolbox - an extraneous element not to be used in completing the task (in red color)

Computational thinking tasks in Matko are organized into five task groups and cover the computational concepts of sequence, algorithms recognition and removal of unnecessary steps, object properties and problem tasks. These concepts were introduced to students in a visual interactive block-based environment, similar to visual programming languages such as Scratch (Resnick, et al., 2009). However, unlike Scratch, due to the participants being first grade primary school students without any experience with similar systems, the blocks the students needed to use were mostly predefined and just needed to be picked out and placed in a correct order or position to form a solution.

Table 1 lists all CT task groups with their respective CT concepts and a representative CT task which is illustrated by two images: one displaying a set of primitives students choose to give their solution and the other showing animation that is being displayed to students whereby they get the visual indication of whether their solution attempt was successful or not.

Table 1
CT task groups with their respective CT concepts and an exemplar CT task

Group	CT concepts	Exemplar CT task and its solution
1	Sequence, algorithm, recognition and removal of unnecessary steps (exemplar task 1.1)	 (Mate)Matko Dodaj vodu Dodaj brašno Dodaj sol Dodaj papriku Stavi tijesto peći na 30 minuta Umijesi tijesto Dodaj vodu
2	Object and its properties (exemplar task 2.2)	 Žaba Broj nogu 2 Tijelo prekriva: Dom: Posebnost zimi: Žaba Žaba ima 2 noge. Tijelo prekriva perje.
3	Problem tasks, Problem tasks with loops (exemplar task 3.1)	 (Mate)Matko Idi gore Idi dolje Idi lijevo Idi desno Upute za zeca: Zeko ide do cilja
4	Problem tasks, Problem tasks with loops (exemplar task 4.5)	 (Mate)Matko Dodaj jedan (1) Dodaj dva (2) Broju tri (3) Dodaj dva (2) Dodaj dva (2) Izraz koji se računa: 5+2=7 Dva Trenutni broj Sedam Rezultat Dodaj sumi dva
5	More complex problem tasks, problem tasks with loops (exemplar task 5.5)	 (Mate)Matko Imate još 0 blok(a). Dodaj tri (3) Dodaj jedan (1) Ponovi tri puta Dodaj dva (2) Stvori broj dvanaest Ponovi tri puta Dodaj tri (3) Dodaj tri (3) Izraz koji se računa: 9+3=12 Tri Trenutni broj Dvanaest Rezultat Dodaj sumi tri

3.3 Prior Experimental Results

The CT tool presented in this paper allowed for detailed data collection of students' usage data for each CT task completed, including both the successful and possibly multiple unsuccessful attempts. The data collected for each student for all five CT task groups included (1) the time students needed to complete a task, (2) the total number of attempts for a task, (3) the number of successful attempts for a task and (4) the number of unsuccessful attempts for a task and (5) the detailed log of the attempt containing all the primitives that a student has chosen in a single task attempt. The analysis of the CT tasks and concepts indicates that the problem tasks and object properties tasks had the largest values of successful and unsuccessful task attempts, with substantial SD observed. SDs both in the case of object properties and problem tasks were high (Boticki, Pivalica & Seow, 2018).

3. Identifying the Most Difficult Computational Thinking Tasks

By drawing on the rich dataset of the collected log data, three key parameters were observed to identify the most difficult task: the total number of unsuccessful attempts per task, the number of students who had at least one unsuccessful attempt per task, and the total amount of time spent on a given task. Upon observing the data and the field notes taken during the experiment, it was concluded that the number of unsuccessful attempts was more relevant in choosing the most difficult CT tasks than the time students spent on a task due to the difficulty of measuring the exact student on-task time. This was due to several reasons including students taking breaks in between the task attempts, submitting the same solution multiple times by accident or because they enjoyed watching the animation appearing after the task evaluation (note that the overall task completion time excludes the time needed to display the animation). Nevertheless, the total amount of time spent on a task was still used as a control factor when choosing the most difficult tasks.

The first task group was excluded from the analysis since students performed extremely well, except for the first task in the first task group (1.1) due to the adjustment period to the tool and the new computational thinking activity. From both the second and third task group, two most difficult tasks were chosen, while in the fourth and fifth only the most difficult task per group was chosen due to the general task similarity (Table 2).

Table 2

Usage statistics of the second task group with the most difficult tasks selected

Task	Total number of unsuccessful attempts	Students with at least an unsuccessful attempt	Average total time spent on the task (seconds)
2.2	127	17	315.62
2.5	98	18	229.25
3.1	46	15	134.79
3.6	44	17	92.98
4.5	35	12	152.34
5.5	32	8	251.72

4. Identifying Patterns in the Most Difficult Computational Thinking Tasks

The selected tasks (2.2, 2.5, 3.1, 3.6, 4.5 and 5.5) are the most difficult tasks for early primary school students according to the selection criteria presented in the previous chapter. Tasks 2.2, 3.1, 4.5 and 5.5 are illustrated in Table 1. In this chapter patterns, or the most common sequence of steps, in completing the tasks are identified. The patterns of correctly completed tasks identify possible common correct solutions attempted by the students, while the patterns of the incorrectly completed tasks allow for the discovery of common misconceptions and errors in early primary learners' computational task completion.

Table 3

Patterns identified in tasks of identifying animal properties (2.2 Frog and 2.5 Rabbit)

Task 2.2 Incorrect Patterns (Frog)		Task 2.5 Incorrect Patterns (Rabbit)	
2 legs, skin, lives in a pond, in winter (*)	56 attempts	4 legs, hair, lives in burrow, in winter (*)	20 attempts
4 legs, skin, lives in a pond, in winter does not sleep	5 attempts	2 legs, hair, lives in haunt, in winter (*)	10 attempts
4 legs, skin and hair, lives in a pond, in winter (*)	3 attempts	2 legs, hair, lives in burrow, in winter (*)	9 attempts
2 legs, hair, lives in a pond, in winter does sleeps	2 attempts	4 legs, hair, lives in burrow, in winter (*)	7 attempts
		4 legs, hair, lives in pond, in winter (*)	2 attempts

In the second task group two most difficult tasks were analyzed for patterns – tasks 2.2 and 2.5. In these tasks, students needed to choose characteristics of an animal from a set of predefined options (number of legs, skin cover type, living area and does it sleep in winter). Table 3 illustrates common misconceptions students encountered in the task completion process. The indication winter sleeps (*) denotes any choice for the sleeping area in winter (sleeps or does not sleep in winter).

Two most difficult tasks were analyzed in the third task group (tasks 3.1 and 3.6). In the task 3.1 students needed to lead a bunny towards the goal using only four simple commands: up, down, left and right. In the task 3.6 students needed to do the same but by choosing more distant goal out of two available goals without reaching the wrong goal in their solution (Table 4).

Table 4

Patterns identified in tasks of leading a bunny towards the goal (3.1 without loops and 3.6 with loops)

Task 3.1 Correct patterns		Task 3.1 Incorrect Patterns		Task 3.6 Correct patterns		Task 3.6 Incorrect Patterns	
up->up->right->right	10 att.	right->up->*	9 att.	right->up->up->left	13 att.	up->up	14 att.
right->right->up->up	8 att.	left->*	4 att.			right->up->up	5 att.
right->up->right->up	3 att,	up->left->*	3 att.				
up->right->up->right	2 att.	up->up->right	2 att.	left->up->up->right	9 att.	left->up->*	5 att.
		up->right->right	2 att.				

In the fourth task group students needed to complete simple mathematics problems using the operations of addition and subtraction. In task 4.5 students were asked to add any number and combination of numbers 1 and 2 to the initial value of 3 in order to reach the value of 7 as the final solution. Correct and incorrect solution patterns for the task 4.5 are given in Table 5.

Table 5

The correct and incorrect patterns in task 4.5

Task 4.5 Correct Patterns			Task 4.5 Incorrect Patterns		
3+	2+2	12 attempts	3+	1+2	8 attempts
			3+	2+1	8 attempts
3+	1+2+1	3 attempts	3+	2+2+2+1	2 attempts
			3+	2+1+2+2	2 attempts

In the fifth task group students needed to solve slightly more complex mathematics problems than in task group 4. An example of such a task is given in the last row of Table 1, where a student needs to produce number 12 by using any combination of values 3, 1 and 2 and a loop of 3 repetitions. Correct and incorrect solution patterns for the task 5.5 are given in Table 6.

Table 6

The correct and incorrect patterns in task 5.5

Task 5.5 Correct Patterns		Task 5.5 Incorrect Patterns	
3*3+3	7 attempts	3*(nothing)+3+3	7 attempts
		3*(3*(….))	4 attempts
3*(3+1)	3 attempts	3*3+2	2 attempts
		3*3+3*(nothing)	2 attempts
3+3*3	2 attempts	3*(nothing)+3+2	2 attempts
		3+2+1	2 attempts

5. Conclusions

When completing the computational thinking tasks with object properties students had to leverage on their prior knowledge in multiple subjects ranging from mathematics, to nature and society and language. Although one of the most common misconception was the wrong recognition of a number of legs a frog or a rabbit has, it is to be noted that students mostly did mistaken legs for arms whereby obvious influence of cartoons they are exposed to on daily basis is detected. Students struggled with language issues when completing the problem of object properties, especially when they needed to specify that rabbit lived in a burrow.

In the tasks students needed to find a path of a bunny to reach the predefined goal by combining the primitives of right, left, up and down, a number of solutions with three steps out of four required steps were identified. These steps would give the correct solution if the students provided an adequate final step. It seems the students used the tool in their trial and error approach facilitated with the animation feature where they could evaluate their solutions visually and continue enhancing their solution by adding additional steps. Additionally, a smaller number of students attempted to “teleport” their bunny from one side to another (for example from top to bottom or from left to right) showing they can generate alternative creative solutions.

The students had no issues when solving simpler mathematical problems as indicated by the analysis of the group 4 tasks. Similar to the task group 3, students used the trial and error approach as well as observed the animated formulas in figuring out intermediary solutions to reach their calculation goal. In the task group 5 children had a lot more misconceptions when loops are used to form the arithmetic expressions, which effectively translates into multiplication.

To sum up, although curriculum contents need to be aligned with the CT task contents to allow student to complete the task more correctly, allowing for an amount of new contents in CT tasks can be beneficial since the students will use the tool to employ trial and error strategies and explore the solution space. By doing that they will iterate through the solution space and identify the correct solution on their own, or with little scaffolds via the computational tool feedback mechanisms. By being open-ended, a CT tool can serve as a polygon for students’ creative solution and lead to creative solutions in the same time engaging students in the problem solution process.

References

Boticki, I; Pivalica, D. & Seow, P. (2018). The Use of Computational Thinking Concepts in Early Primary School. Proceedings of the International Conference of Computational Thinking Education 2018 (CTE2018). *In publication*.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking.

Cooper, S., Perez, L., & Rainey, D. (2010). K-12 Computational Learning. *Communications of the Acm, 53*(11), 27-29.

Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play. In C. Kazimoglu, M. Kiernan, L. Bacon, L. MacKinnon, H. Ali, Y. Shi, D. Khazanchi, M. Lees, G. VanAlbada, J. Dongarra, & P. Sloot (Eds.), *Proceedings of the International Conference on Computational Science, Iccs 2012* (Vol. 9, pp. 522-531).

Lye, S., & Koh, J. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51-61.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., Kafai, Y. (2009, 11). Scratch: Programming for All. *Communications of the ACM, 52*(11), 60-67.

Roman-Gonzalez, M., Perez-Gonzalez, J.-C., & Jimenez-Fernandez, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678-691.

Wilson, A., & Moffat, D. C. (2010). Evaluating Scratch to introduce younger schoolchildren to programming. Proceedings of the 22nd Annual Workshop of the Psychology of Programming Interest Group, 64-75

Wing, J. (2006). Computational thinking. *Communications of the Acm, 49*(3), 33-35.

Wing, J. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society a-Mathematical Physical and Engineering Sciences, 366*(1881), 3717-3725.

Wing, J. (2011). Research Notebook: Computational Thinking--What and Why? *The Link - The Magazine of the Varnegie Mellon University School of Computer Science*.